МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ

Національний технічний університет
«Харківський політехнічний інститут»

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторної роботи
«Функції у програмах мовою Delphi»

з курсу «Програмування»
для студентів напряму 6.040302 – Інформатика
(спеціалізація «Соціальна інформатика»)

Затверджено редакційно-видавничою
радою університету,
протокол № 2 від 01.12.10.

Харків НТУ «ХПІ» 2011

Методичні вказівки до лабораторної роботи «Функції у програмах мовою Delphi» з курсу «Програмування» для студентів напряму 6.040302 – Інформатика (спеціалізація «Соціальна інформатика») / Уклад. М. І. Безменов. – Х. : НТУ «ХПІ», 2011. – 16 с.

Укладач М. І. Безменов

Рецензент Л. М. Любчик

Кафедра системного аналізу і управління

ВСТУП

Одним із засобів, використання яких полегшує розробку більш менш складних программ, є підпрограми, зокрема, – функції.

Метою даної лабораторної роботи є освоєння методики визначення та практичного застосування функцій у програмах, написаних мовою Delphi.

1. ТЕОРЕТИЧНІ ОСНОВИ

1.1. Загальне призначення підпрограм

Підпрограми – це фрагменти програми, оформлені спеціальним способом і призначені для розв'язання конкретних задач.

Крім підпрограм, що входять у бібліотеки, які можна підключати до розроблюваної програми, програміст має можливість розробляти і використовувати власні підпрограми. Підпрограма оформлюється у програмі один раз[1)], а використовується багаторазово у різних місцях прогарами, виконуючи одні й ті ж самі дії над різними оброблюваними даними. Підпрограми дозволяють структурувати програму; написання великих програм без використання підпрограм говорить про поганий стиль програмування. Програма не повинна залежати від особливостей реалізації підпрограм – підпрограми слід писати так, щоб їх можна було переносити з одних програм в інші і створювати бібліотеки підпрограм.

У Delphi передбачено підпрограми-процедури і підпрограми-функції. При звертанні до підпрограми зазначають її ім'я, а в дужках через кому перераховують параметри, від яких залежить результат обчислення (наприклад, щоб обчислити синус деякого значення, це значення вказується як параметр).

Опрацьовувачі подій також є підпрограмами, але їх особливістю є те, що вини викликаються на виконання автоматично, хоч не є забороненим і їх примусовий виклик на виконання. Існує ще два види підпрограм – конструктори та деструктори, які визначаються спеціальним чином у класах.

Використання підпрограм дає такі дві основних переваги:

1) відокремлюється хід розв'язання від конкретної реалізації підпрограми – немає потреби знати, як реалізована підпрограма, потрібно тільки вміти до неї звернутися;

[1)] Сказане не стосується підпрограм, визначених всередині інших підпрограм або в різних модулях. Однак наявність декількох визначень однакових підпрограм у одній і тій самій програмі є свідченням поганого стилю програмування.

2) опис підпрограми в програмі дається один раз, а використовувати її можна багаторазово.

1.2. Функції

Нестрогим математичним визначенням поняття «функція» можна прийняти таке визначення.

Функція – це закон, що визначає однозначну відповідність кожному елементу деякої множини *X* один елемент деякої множини *Y*, причому елементи множини *X* у свою чергу можуть бути множинами.

У Delphi функція відповідає наведеному вище математичному визначенню, а саме, це оформлений спеціальним чином фрагмент програми, що перетворює одне або кілька значень не обов'язково того самого типу в одне значення деякого типу.

Функція визначається у відповідності з наступним форматом:

Заголовок функції

Тіло функції

Розглянемо особливості опису функції і звертання до неї.

1. Заголовок функції починається зі службового слова **`function`**, зразу за яким йде ім'я функції. Після імені функції в круглих дужках зазначаються формальні параметри та їхні типи. Якщо параметрів декілька, то вони оголошуються через крапку з комою. Якщо кілька параметрів мають один і той самий тип, то ці параметри можна подати через кому з вказівкою спільного типу. Після дужок, у які укладені формальні параметри, через двокрапку вказують тип функції (тип значення, що повертається), після якого ставиться роздільник «крапка з комою». У праматері Delphi – мові Turbo Pascal функція служила для обчислення тільки одного значення, що передавалося в програму через ім'я функції, внаслідок чого типом значення, що поверталося, міг бути будь-який скалярний тип (числовий, символьний, булівський, перелічений), рядковий тип або вказівник. У Delphi типом функції може бути довільний тип (крім файлових). Наприклад:

```
function fun(a, b, c: Real; d: Integer): Real;
```

2. Тіло функції в загальному випадку складається з двох частин – розділу описів, який може бути відсутнім, та виконуваної частини. Остання починається зі службового слова **`begin`** і завершується службовим словом **`end`**, за яким ставиться роздільник «крапка з комою». Змінні, типи, константи тощо, оголошені у тілі функції перед її виконуваною частиною, які називаються

локальними. Ці змінні поза функцією не існують, оскільки під них виділяється пам'ять щоразу при виклику функції, а при виході з функції ці змінні знищуються.

```
function fun(a, b, c: Real; d: Integer): Real;
var
  g: Real;
begin

end;
```

3. Тип значення, що повертається, може бути заданий тільки ім'ям. Не можна, наприклад, вказати тип **array**`[1..20]` **of** `Real` – у цьому випадку у програмі слід ввести новий тип і використати його при зазначенні типу функції (наприклад, **type** `TReal20 =` **array**`[1..20]` **of** `Real`.

4. Тип формальних параметрів теж задається ім'ям (можливе також застосування типу «відкритий масив»; наприклад, **array of** `Real`). За необхідності використання як параметрів масивів, записів та інших елементів даних, типи яких не задаються одним ім'ям, у розділі опису типів визначають користувальницькі типи даних (див. попередній пункт).

5. Формальні параметри є локальними змінними.

6. Імена локальних змінних не можуть збігатися з іменами формальних параметрів (це є наслідком з попереднього).

7. Якщо функція не має формальних параметрів, то круглі дужки в заголовку не ставляться (в Delphi дозволено ставити круглі дужки й у цьому випадку).

8. Деякі з параметрів можуть не мати типу.

9. Результат роботи функції повертається в точку виклику тільки якщо у функції він буде записаний у внутрішню змінну `Result`, яку має будь-яка функція і яка завжди має тип, що збігається з типом функції, – інакше результат роботи функції не буде переданий у точку виклику. Змінну `Result` спеціально **описувати не потрібно**.

10. Результат роботи функції передається в точку виклику і через так звані **var**- та **out**-параметри, але такі параметри частіше притаманні процедурам.

11. Всі змінні, описані в модулі до оголошення функції, також відомі усередині функції. Ці змінні називаються ***глобальними***. У зв'язку з загальнодоступністю глобальних змінних у складних випадках можлива ситуація, коли одна функція може змінити значення змінної непомітно для іншої, результатом чого стає поява помилок, які досить важко виявляються. Щоб уник-

нути цього, рекомендується обмежувати використання глобальних змінних. Глобальні змінні необхідні в тому випадку, коли програмісту потрібно зробити дані доступними для декількох функцій, а передача даних з однієї функції в іншу є проблематичною.

12. Імена формальних параметрів і локальних змінних можуть збігатися з різними іменами, оголошеними поза функцією. У цьому випадку в тілі функції активними будуть імена, оголошені в ній.

13. Функція оголошується в розділі описів. Її можна оголосити як у секції **`implementation`** модуля (у цьому випадку вона буде відома всюди нижче точки опису), так і у розділі опису будь-якої іншої підпрограми (правда, у цьому випадку вона буде відома тільки усередині цієї підпрограми).

14. Функція може бути оголошена також і в секції **`interface`** модуля. У цьому випадку в секції **`interface`** оголошується тільки заголовок функції, а в секції **`implementation`** дається її повне визначення. Такий спосіб опису функції робить її відомою у всіх модулях, до яких підключений даний модуль.

15. Функції можуть бути описані й у розділі описів проекту, але доступ до них у такому разі можливий тільки за допомогою операторів, записаних у виконуваній частині проекту. Оскільки програміст змінює проект дуже рідко, подібного роду опис є практично безглуздим.

16. Для звертання до функції необхідно згадати її ім’я в якому-небудь операторі. При цьому замість формальних параметрів повинні бути підставлені фактичні параметри, якими можуть виступати змінні, константи, вирази. Фактичні параметри розділяються комами. Типи формальних і фактичних параметрів мають бути сумісними за присвоюванням. Якщо функція не має параметрів, то при звертанні до неї дужки можна не вказувати. Приклад звертання:

```
z := Sqr(fun(a, x, 1.5, 2 * n)) - 0.2;
```

17. Якщо в програмі активізована директива компілятору `$X` (стан `{$X+}` – розширений синтаксис), то до функції можна звернутися як до окремого оператора (наприклад, `factorial(10);`). Однак це можна робити тільки у випадках, коли значення, що повертається, не використовується. За умовчанням установлена директива `{$X+}`.

2. ПРИКЛАДИ ПРОГРАМ

Приклад 1. Дано дійсні числа *v* і *w*. Обчислити значення виразу

$$\mathrm{sh}\,(v+\cos(w+1.5))+\mathrm{ch}\,(v+\sin(w+1.5))-e^{\mathrm{sh}\,2+\mathrm{ch}\,\pi},$$

де $\mathrm{sh}\, z$, $\mathrm{ch}\, z$ – відповідно гіперболічний синус і гіперболічний косинус числа z, які визначаються такими формулами:

$$\mathrm{sh}\, z = \frac{e^z - e^{-z}}{2}, \quad \mathrm{ch}\, z = \frac{e^z + e^{-z}}{2}.$$

Визначити функції обчислення гіперболічного синуса і гіперболічного косинуса.

Розв'язання. Розмістимо на формі два редактори Edit1 та Edit2 для введення чисел *v* і *w* відповідно, багаторядковий редактор Memo1 для виведення результату та кнопку Button1 та дві мітка, розмістивши їх над редакторами. Запишемо у властивості Text редакторів число 0, у властивість Caption мітки над редактором Edit1 підказку Уведіть v, а у властивість Caption мітки над редактором Edit21 підказку Уведіть w . В такому разі задачу розв'язує такий оброблювач події OnClick кнопки Button1:

```
function Sh(z: Real): Real;              { Визначення функції }
begin
  Result := (Exp(z) - Exp(-z)) / 2;
end;

function Ch(z: Real): Real;              { Визначення функції }
begin
  Result := (Exp(z) + Exp(-z)) / 2;
end;

procedure TForm1.Button1Click(Sender: TObject);
var
  v, w: Real;
begin
  DecimalSeparator := '.';
  v := StrToInt(Edit1.Text); w := StrToInt(Edit2.Text);
  Memo1.Lines.Add('Value of expression is equal ' +
                  FloatToStr(Sh(v + Cos(w + 1.5)) +
                  Ch(v + Sin(w + 1.5)) - Exp(Sh(2) + Ch(Pi))));
end;
```

Приклад 2. Дано *n* об'єктів, $n \le 12$. Скільки можна сформувати з них різних сполучень по *m* ($m \le n$) об'єктів. Число сполучень з *n* по *m* визначається формулою

$$C_n^m = \frac{n!}{(n-m)!\cdot m!}.$$

Розв'язання. Скористаємося формою з прикладу 1, видаливши з неї редактор Edit2 та відповідну мітку. Запишемо у властивість Caption мітки, що залишилася текст Уведіть N (N <= 12), у властивості Text редактора число 1, а у властивість Caption кнопки Button1 підказку Уведіть. Оголосимо також в секції **public** опису класу TForm1 поля N, M і CNM типу Integer. Крім того, у секції **implementation** опишемо функцію:

```
function factorial(k: Integer): Integer;
var
  i: Integer;
begin
  Result := 1;
  for i := 1 to k do
    Result := Result * i;
end;
```

Створимо також наступний опрацьовувач події OnClick для компонента Button1:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  Edit1.SetFocus;
  if Tag = 0 then begin
    N := StrToInt(Edit1.Text);
    Label1.Caption := 'Уведіть M (M <= N)';
  end
  else begin
    M := StrToInt(Edit1.Text);
    CNM := factorial(N) div factorial(N - M) div factorial(M);
    Memo1.Lines.Add('Число сполучень з ' +
                        IntToStr(N) + ' по ' + IntToStr(M) +
                        ' дорівнює ' + IntToStr(CNM));
    Button1.Visible := False;
    Edit1.Visible := False;
    Label1.Visible := False;
end;
  Tag := Tag + 1;
end;
```

Примітка. Властивість Tag форми, що за умовчанням має значення 0, використовується для як перемикач. Якщо Tag = 0, уводиться значення змінної N, інакше – уводиться змінна M та обчислюється результат.

Приклад 3. Обчислити слід (суму елементів, розташованих на головній діагоналі) квадратної матриці розміру не більш ніж 20×20. Визначити функцію обчислення сліду квадратної матриці.

Розв'язання. Скористаємося формою з прикладу 2, внісши такі зміни в значення властивостей:

Мітка:

Name — lbOutput1

Caption — Уведіть розмірність матриці (N<=20)

Однорядковий редактор:

Name — edInput1

Багаторядкове поле:

Name — mmOutput1

Lines — очистимо

У секції **interface** модуля перед описом класу TForm1 визначимо тип для наступного оголошення двовимірного масиву:

```
type
  T20x20 = array[1..20, 1..20] of Real;
```

У визначення ж типу TForm1 в секції **public** додамо опис трьох полів:

```
N: Byte;
matrix: T20x20;
sum: Real;
```

Крім того, перед кодом процедури TForm1.Button1Click опишемо таку функцію:

```
function TraceOfMatr(a: T20x20; N: Byte): Real;     //Заголовок
var
  i: Byte;
  sum: Real;
begin
  sum := 0;
  for i := 1 to N do
    sum := sum + a[i, i];
  Result := sum;         //Результат заноситься в змінну Result
end;                                                //Кінець функції
```

Для розв'язання задачі скористаємося такими двома опрацьовувачами подій:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
```

```
  DecimalSeparator := '.';
  edInput1.Text := '1';
  lbOutput1.Caption := 'Уведіть розмірність матриці (N<=20)';
  Button1.Caption := 'Виконати';
end;

procedure TForm1.Button1Click(Sender: TObject);
var
  i, j: Integer;
begin
  N := StrToInt(Trim(edInput1.Text));
  for i := 1 to N do
    for j := 1 to N do begin
       matrix[i, j] := StrToInt(
            InputBox('Уведення матриці',
                     'Уведіть a[' + IntToStr(i) +
                     ', ' + IntToStr(j) + ']', '0'));
    end;
  sum := TraceOfMatr(matrix, N);    //При звертанні до функції
           //замість формальних параметрів a та N підставлені
                                 //фактичні параметри matrix та N
  mmOutput1.Lines.Add('Слід матриці дорівнює ' +
                          FloatToStr(sum));
  Button1.Visible := False;
  lbOutput1.Visible := False;
  edInput1.Visible := False;
end;
```

Оскільки в список формальних параметрів функції не можна включити опис

```
a: array[1..20, 1..20] of Real;
```

у програмі оголошений тип `T20x20`, що використаний як при описі поля `matrix` класу `TForm1`, так і при описі параметра `a` функції `TraceOfMatr` з метою забезпечення еквівалентності типів формальних і фактичних параметрів.

3. ЗАВДАННЯ НА ЛАБОРАТОРНУ РОБОТУ

За час, відведений для виконання лабораторної роботи (2 академічні години), студент повинен:

1. Розробити алгоритм розв'язання задачі, запропонованої для програмування.
2. Здійснити проектування форми для функціонування розроблюваної програми.
3. Здійснити програмну реалізацію розробленого алгоритму.
4. Здійснити відлагодження програми, виправивши синтаксичні та логічні помилки.
5. Підібрати тестові дані для перевірки програми, включаючи виняткові випадки.
6. Оформити звіт до лабораторної роботи.
7. Відповісти на контрольні запитання.
8. Здати викладачу працездатну програму з демонстрацією її роботи на декількох варіантах вихідних даних.

4. ВАРІАНТИ ЗАДАЧ

1. Дано натуральне число n. Одержати всі піфагорові трійки натуральних чисел, кожне з яких не перевищує n, тобто всі такі трійки натуральних чисел a, b, c, що $a^2 + b^2 = c^2$ $(a \le b \le c \le n)$. Оформити функцію перевірки того, що дана трійка натуральних чисел є піфагоровою.
2. Дано натуральне число n, дійсні числа a_0, a_1, …, a_n, x. Обчислити $a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$. Визначити функцію піднесення дійсного значення до цілого невід'ємного степеня.
3. Для дійсних значень x з інтервалу $-1 < x < 1$ справедливе співвідношення

$$\operatorname{arctg} x = x - \frac{x^3}{3} + \frac{x^5}{5} - \ldots$$

Визначити функцію з двома параметрами для обчислення значення $\operatorname{arctg} x$ за наведеною вище формулою з урахуванням усіх доданків, що за модулем більші від заданого додатного значення ε.

4. Створити власні функції, призначені для обчислення наближених значень функцій e^x, $\sin x$, $\cos x$ за допомогою розкладання в нескінченні ряди:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots,$$

$$\sin x = x - \frac{x3}{3!} + \frac{x^5}{5!} - \ldots,$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots$$

У сумах ураховувати всі доданки, що за модулем більші від заданого додатного значення ε.

Дано дійсні числа *x*, μ (μ > 0). Послідовно розглядаючи як ε числа з послідовності 0.5, 0.25, 0.125, …, для кожної з трьох визначених у програмі функцій знайти таке значення ε, за якого значення цієї функції вперше буде відхилятися від значення, обчисленого за допомогою відповідної функції з бібліотеки стандартних програм обраної мови програмування, на величину, що не перевищує μ.

5. Дано ціле число *n*. Знайти найменше *число Сміта*, що перевищує *n*. Визначити функцію перевірки цілого числа на те, що воно є числом Сміта, а також функцію перевірки цілого числа на те, що воно є простим.

Примітка. Натуральне число називається числом Сміта, якщо воно не є простим і при цьому сума його цифр дорівнює сумі цифр розкладання цього числа на прості множники. Число 4937775 розкладається на прості множники в такий спосіб: 4937775 = 3 × 5 × 5 × 65837. Сума цифр цього розкладання дорівнює 3 + 5 + 5 + 6 + 5 + 8 + 3 + 7 = 42. Цьому ж числу дорівнює сума цифр самого числа 4937775: 4 + 9 + 3 + 7 + 7 + 7 + 5 = 42.

6. Дано дійсні числа *a*, *b*, *c*, *d*. Знайти площу шестикутника, зображеного на рис. 4.1. Визначити підпрограму обчислення площі трикутника за трьома його сторонами, заданими дійсними числами. Якщо ці числа не дозволяють побудувати трикутник, то результатом роботи підпрограми має бути значення –1.

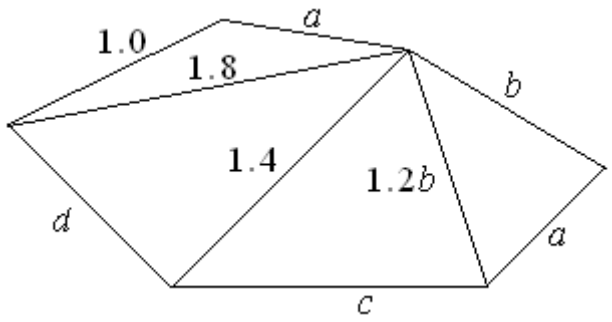

Рис. 4.1

7. Два натуральних числа називаються *дружніми*, якщо кожне з них дорівнює сумі всіх дільників іншого, крім самого цього числа. Дано натуральні числа

m, n ($m < n$). Знайти всі пари дружніх чисел у діапазоні від m до n. Визначити функцію перевірки того, що два натуральних числа є дружніми.

8. Дано натуральне число n, цілі числа a_1, a_2, …, a_n. Знайти всі відрізки послідовності a_1, a_2, …, a_n з членів, які йдуть підряд і складаються з довершених чисел. Визначити підпрограму, що дозволяє розпізнавати довершені числа. **Примітка.** *Довершеним* називають таке натуральне число, що дорівнює сумі всіх своїх дільників, менших від самого числа ($6 = 1 + 2 + 3$).
9. Дано натуральне число n. Для $k = 1, 2, \ldots, n$ одержати суму тих чисел виду $k^3 + (3k + 1) \cdot n$, які є потроєними непарними. Визначити функцію перевірки того, що натуральне число є потроєне непарне.
10. Дано натуральне число n, цілі числа a_1, a_2, …, a_n. Для послідовності a_1, a_2, …, a_n розглянути підпослідовності членів, що йдуть підряд і складаються:
    а) з повних квадратів;
    б) із степенів числа 5;
    в) з простих чисел.

    У кожному випадку одержати найбільшу з довжин розглянутих відрізків. Визначити підпрограми, що дозволяють розпізнавати повні квадрати, степені числа 5, прості числа.
11. Дано два цілочислових одновимірних масиви однакового розміру. Знайти позицію першого елемента, яким відрізняються ці масиви. Описати підпрограму визначення номера першого елемента, яким відрізняються будь-які два цілочислових одновимірних масиви з однаковою розмірністю. Якщо масиви еквівалентні, то результатом роботи підпрограми має бути значення –1.
12. Розробити функцію, що дозволяє визначити порядковий номер крайнього праворуч входження заданого цілого числа c у одновимірний цілочисловий масив, що є її параметром. Якщо масив не містить число c, результатом роботи функції має бути значення –1.
13. Розробити функцію, що дозволяє визначити порядковий номер першого входження в заданий одновимірний цілочисловий масив елементів другого одновимірного цілочислового масиву (масиви не обов'язково повинні мати один і той самий розмір). Результатом роботи функції має бути значення –1, якщо перший масив не містить жодного числа, що належить і другому заданому масиву.
14. Описати функцію, що обчислює суму елементів k-го стовпця дійсної матриці $\mathbf{A} = (a_{ij})_{mn}$ розміром $m \times n$. Для даної дійсної матриці $\mathbf{A}$ розміром

$m \times n$, використовуючи функцію підсумовування елементів рядка матриці, знайти номер рядка з мінімальною сумою елементів.

15. Дано натуральне число *n*. З'ясувати, чи є серед чисел *n*, $n+1$, …, $2n$ близнюки, тобто прості числа, різниця між якими дорівнює 2. Визначити підпрограму, що дозволяє розпізнавати прості числа.
16. Натуральне число з *n* цифр є *числом Армстронга*, якщо сума його цифр, зведених до *n*-го степеня, дорівнює самому числу (наприклад, $153 = 1^3 + 5^3 + 3^3$). Оформити підпрограму перевірки того, що натуральне число є числом Армстронга, і скористатися нею для одержання всіх дво-, три-, чотири- і п'ятицифрових чисел Армстронга.
17. Дано парне число $n > 2$. Перевірити для цього числа *гіпотезу Гольдбаха*, яка полягає в тому, що кожне парне *n*, більше 2, подається у вигляді суми двох простих чисел. Визначити підпрограму, що дозволяє розпізнавати прості числа.
18. Будемо говорити, що натуральне число *v* «краще» за інше натуральне число *w*, якщо сума цифр *v* більша від суми цифр *w*, а в разі рівності сум їхніх цифр *v* кращім буде тоді, коли $v < w$. Наприклад, число 124 краще від числа 123, тому що в першого з них сума цифр дорівнює семи, а в другого – шести. Водночас число 3 краще за 111, тому що в них рівні суми цифр, але $3 < 111$.

    Дано натуральне число *n*. Знайти такий дільник цього числа, що є кращим за будь-який інший дільник. Визначити функцію знаходження найкращого дільника натурального числа, заданого як її параметр.
19. Дано натуральне число *n* і масив *a* з *n* натуральних чисел. Одержати масив *b* з «найкращих» дільників масиву *a*. Визначити три функції – функцію обчислення суми цифр цілого числа, функцію перевірки того, що одне натуральне число «краще» від іншого, і функцію відшукання «найкращого» дільника натурального числа.

5. КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що забезпечує використання підпрограм?
2. Як у програмі визначається функція? Опишіть її формат.
3. Як визначається заголовок функції?
4. Що таке тіло функції?
5. Що таке формальні параметри і як вони описуються?
6. Що таке локальні змінні й у чому їхня особливість?

7. Що таке глобальні змінні та чим зумовлені рекомендації щодо обмеження їх використання?
8. Як описати масив – формальний параметр функції?
9. Чи може функція не мати параметрів?
10. Яке обмеження існує на імена локальних змінних?
11. Чи можуть імена глобальних змінних співпадати з іменами формальних параметрів та локальних змінних?
12. Як результат роботи функції передається назовні?
13. Як здійснюється звертання до функції?
14. Чи може бути функція оголошена всередині іншої функції?
15. Що таке формальні і фактичні параметри?

СПИСОК ЛІТЕРАТУРИ

1. Безменов, М. І. Основи програмування в середовищі Delphi : навч. посіб. / М. І. Безменов. – Х. : НТУ «ХПІ», 2010. – 608 с.
2. Кэнту, М. Delphi 7 : Для профессионалов / М. Кэнту – СПб. : Питер, 2004. – 1101 с.
3. Архангельский, А. Я. Программирование в Delphi 6 / А. Я. Архангельский. – М. : БИНОМ, 2002. – 1120 с.
4. Дарахвелидзе, П. Г. Программирование в Delphi 7 / П. Г. Дарахвелидзе, Е. П. Марков. – СПб. : БХВ-Петербург, 2003. – 784 с.
5. Культин, Н. Б. Основы программирования в Delphi 7 / Н. Б. Культин. – СПб. : БХВ-Петербург, 2003. – 608 с.
6. Пестриков, В. М. Delphi на примерах / В. М. Пестриков, А. Н. Маслобоев. – СПб. : БХВ-Петербург, 2005. – 496 с.
7. Ремкеев, А. А. Курс Delphi для начинающих. Полигон нестандартных задач / А. А. Ремкеев, С. В. Федотова. – М. : СОЛОН-Пресс, 2006. – 360 с.
8. Митчелл, К. Керман. Программирование и отладка в Delphi : учебный курс / Митчелл К. Керман. – М. : Вильямс, 2004. – 720 с.
9. Парижский, С. М. Delphi : Только практика / С. М. Парижский. – К. : МК-Пресс, 2005. – 208 с.
10. Культин, Н. Б. Основы программирования в Delphi 2007 / Н. Б. Культин. – СПб. : БХВ-Петербург, 2008. – 480 с.

Навчальне видання

Методичні вказівки
до лабораторної роботи
«Функції у програмах мовою Delphi»
з курсу «Програмування» для студентів напряму 6.040302 – Інформатика
(спеціалізація «Соціальна інформатика»)

Укладач БЕЗМЕНОВ Микола Іванович

Відповідальний за випуск О. С. Куценко
Роботу до видання рекомендував О. В. Горелий

За авторською редакцією

План 2011 р., поз. 3 / 77-11

Підп. до друку 23.05.2011 р. Формат 60 × 84 $^{1}/_{16}$. Папір офісний.
Riso-друк. Гарнітура Таймс. Ум. друк. арк. 0,9. Наклад 50 прим.
Зам. № 159. Ціна договірна.

Видавничий центр НТУ «ХПІ».
Свідоцтво про державну реєстрацію ДК № 3657 від 24.12.2009 р.
61002, Харків, вул. Фрунзе, 21

Друкарня НТУ «ХПІ», 61002, Харків, вул. Фрунзе, 21